Final Report for ECE 445, Senior Design, Spring 2017
TA: Kexin Hui

May 2017
Project #74

Jie Jia
Yifei Li
Zhengfeng Wu

OmniMouse

Abstract

The following paper details the thought process and build behind the OmniMouse Capacitive Sensor HID device. The mouse is built using eight round capacitive sensors that are placed around the inside of an acrylic dome that are connected to a microcontroller. The device has both wired and wireless operation, as well as button input. It is battery operated, lasting 18.75 hours before charging (via USB) is needed. The mouse aims to be an ergonomic replacement for standard computer interface devices.

Table of Contents

Objective

Computing technologies has become an important aspect of daily life for many people across the world, an estimated 3.5 billion people are connected to the internet [1], and there are likely many more who have access to a computer that is not connected. For those using devices such as desktop or laptop computers, one of the most common ways to interface with a computer is through the use of a mouse. The operating a standard mouse crosses the user's ulna and radius bones and requires repetitive motions such as clicking and moving the device to give input. Over time, these movements can lead to discomfort or permanent damage. This is a result of the repetitive motions that the user undergoes, which leads to strain and microscopic tears throughout the utilized regions [2]. Research has suggested that to minimize these kinds of injuries, one should minimize wrist movement, offload strain to multiple locations, have a light grip, and avoiding positions that prevent proper circulation [3].

Our goal is to create a mouse that minimizes strain and injury for the user through varied non-repetitive motions and lessened hand pressures. We plan to address the various points of stress for using a mouse individually to create a final product that will hopefully help prevent and reduce the prevalence of stress due to computer mouse use.

Background

The main issues that we want to address are the minimization of wrist movement, strain being offloaded to multiple locations, the use of a light grip, and providing a position that promotes proper circulation. Our design is a stationary mouse with capacitive sensors around the base of the mouse, allowing the user to keep their hand in one position, minimizing any wrist movement. Since the mouse is stationary, little grip from the user is needed, further lowering strain on the user's hand. Movement input is done through the capacitive sensors detecting how close the user's hand is, and through the data of multiple sensors, we can map the location of the hand on the mouse. Using this data lets us calculate what direction to move the mouse cursor. This offloads strain to multiple locations since every direction the user goes can be controlled with a different part of the hand. The angle and shape of the mouse will be optimized to provide the user with a neutral resting position while using the device, which allows for the best circulation within the whole arm.

High Level Requirements

1. The mouse should be ergonomic and shouldn't strain the wrist or any other part of the body.

2. Function like a standard laser based mouse, have a minimum polling rate of 60hz,, work on any type of surface, and have a battery life of at least 4 hours.

3. The use of the mouse should be intuitive. The mouse should have basic functions like "moving", "clicking" and "scrolling".

Block Diagram

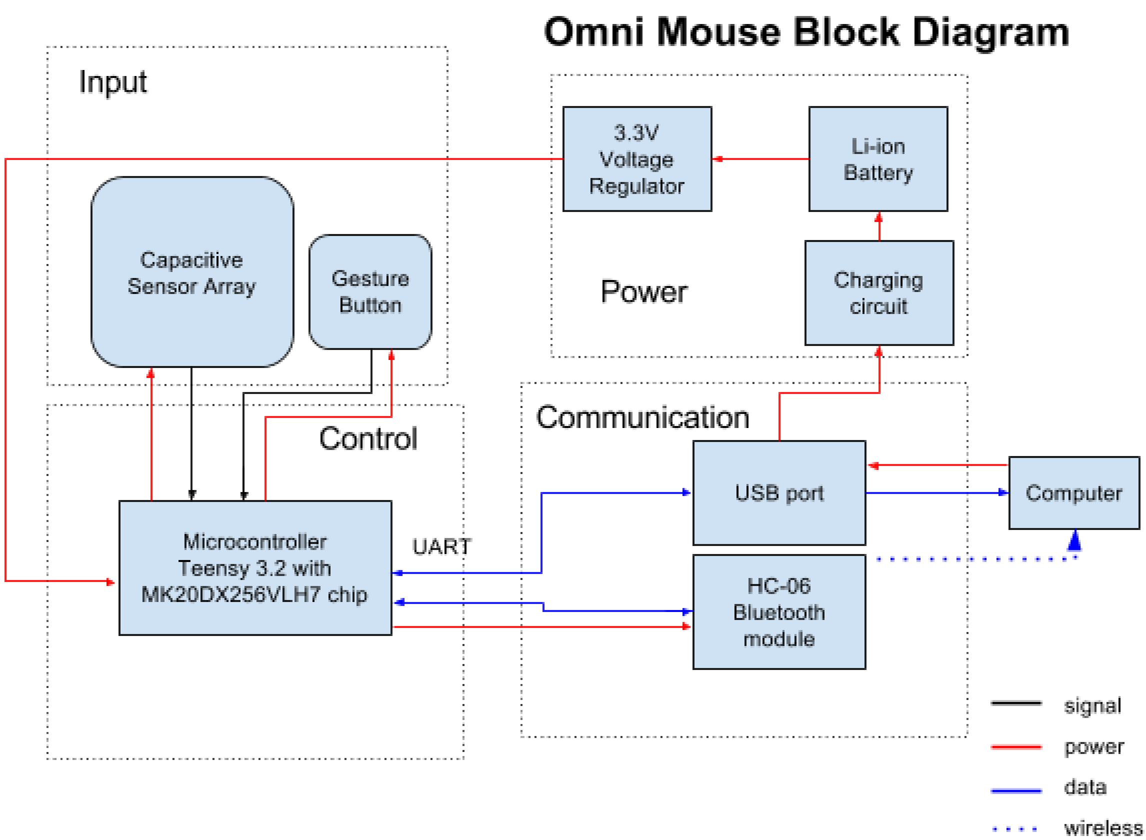


Figure 1: OmniMouse Block Diagram

Diagram Description

In our design, there are four major components to be implemented: Power, Control, Connection and Input. The power supply and power circuit provide power to our system, through a lithium ion battery. This battery is rechargeable through USB power. When the USB is plugged into a compatible computer, it will both recharge and act as a wired mouse. Our control unit is our microcontroller. Our Teensy 3.2 will handle all the IO and data processing needed to transmit data to our mouse driver on the connected computer. The Input section is designed to track hand movement of the user. Our algorithm will process data from this sensor array and translate it into a two dimensional plane for our mouse driver. Finally, the communication module handles wired and wireless connections. Wireless connection will be done using Bluetooth LE, and the wired connection will be through USB OTG.

Control Flow

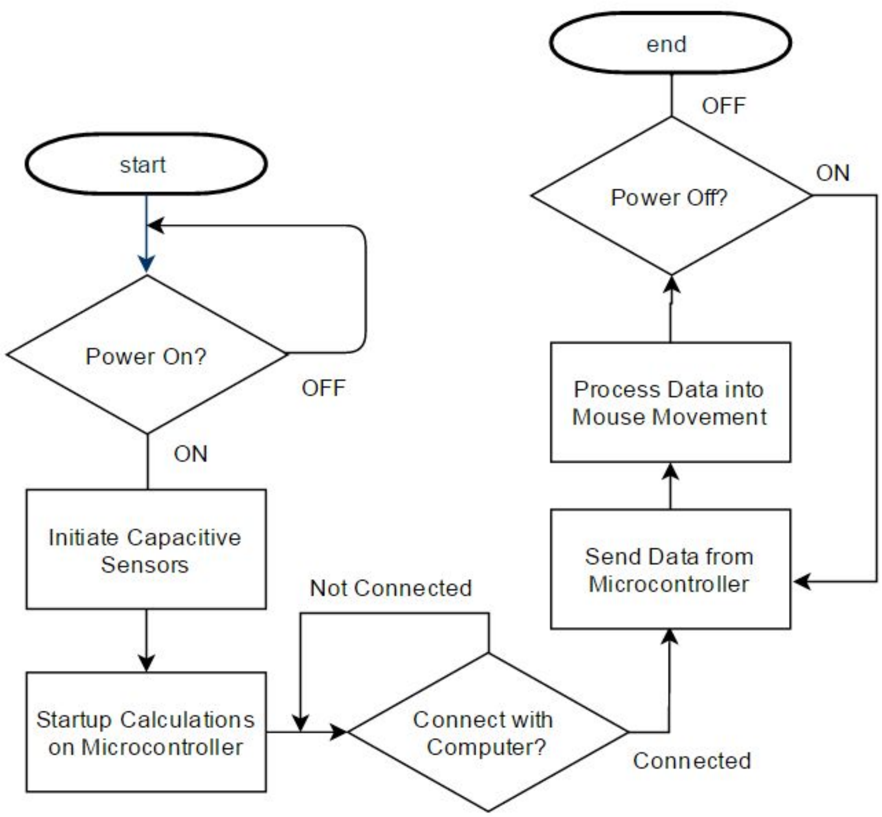


Figure 2: Microcontroller and Driver Flow Diagram

Diagram Description:

This diagram depicts the process in which data is obtained from the sensors of the device and sent to the connected device. Upon mouse startup, initial calculations are done to tune the accuracy of the sensors. This calibration calculates the ambient values that the mouse records and normalizes them so we get consistent and comparable values from our sensors. After these are complete, the mouse is free to connect with a device either via Bluetooth or USB, after which it will continuously send the sensor data packets to be processed by the computer driver.

Circuit Schematic

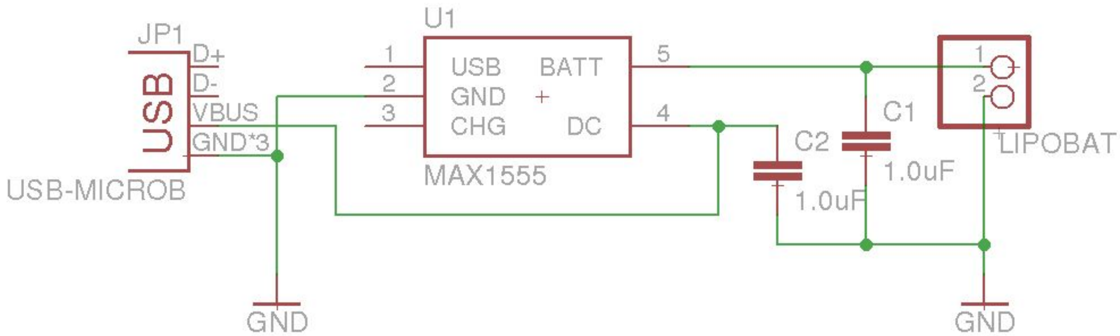


Figure 3: USB Charging Circuit [4]

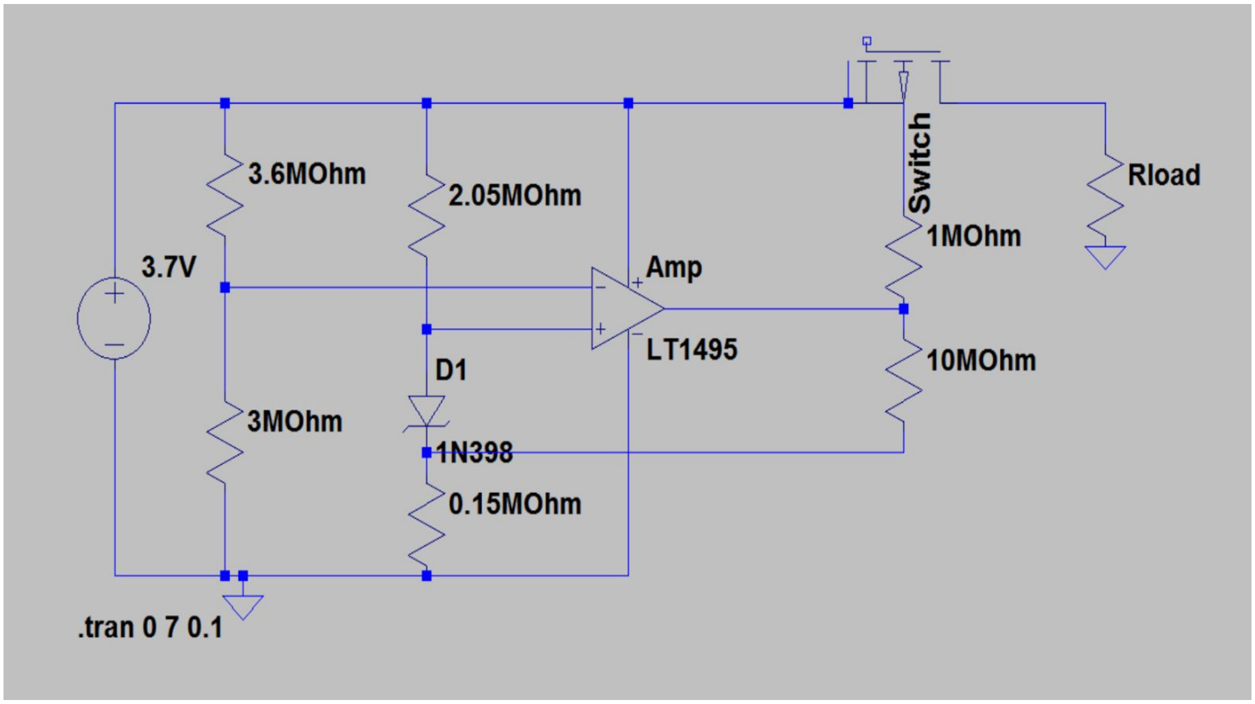
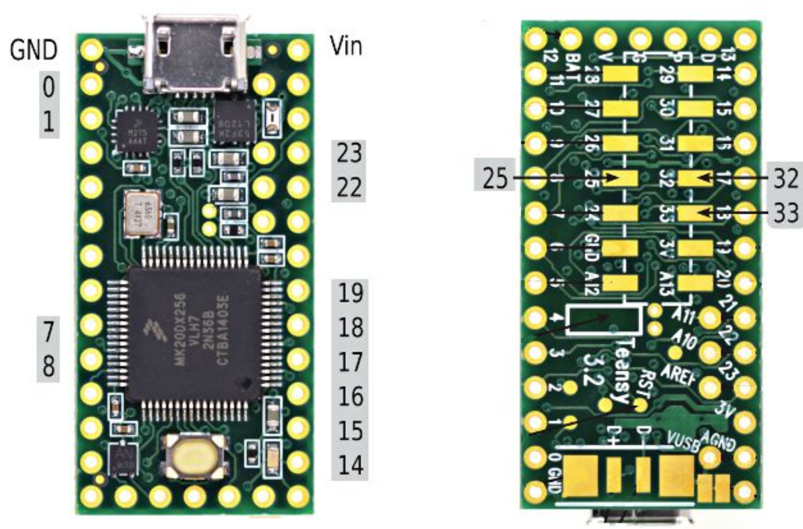


Figure 4: Power Protection Circuit

Teensy 3.2 Pinout



Pins: 0, 1, 14, ,15, 16, 17, 18, 19, 22, 23, 25, 32, 33 - Capacitive Sensors
Pins: 7, 8 - RX, TX for Bluetooth Module
GND: Ground
Vin: Power Block (Battery)

Figure 5: Teensy 3.2 Pinout [5]

Design Overview (Theoretical)

I. Input

Our input consists of the capacitive sensor array and a tactile gesture button. The sensor array consists eight copper discs positioned circularly within our mouse dome. This array takes input from the user and sends it to the microcontrollers, which process the input into mouse movements data packets that can be sent to the computer to be processed by the mouse driver. The gesture button (Omron B3F-4055 2.5N Tactile Switch) will be located on the underside of the mouse, and will be actuated to enable the use of “gestures”. These gestures will require the user to click down on the gesture button (2.5N actuation force) and move their hand to one of the sides. This will activate mouse inputs such as click, right click, and scroll. Double click can be achieved through two normal left clicks within an allotted time. Middle clicking (like to close a tab, or to scroll) can be done by activating both left and right click at the same time. The sensors were placed radially around the dome of the device to give each sensor an area to control.

II. Power

Our power circuit consists of an li-ion battery, a charging circuit, and a safety circuit. The power block consists of our battery, which is used to power the device while in wireless mode, and charges when it is in wired mode. The charging circuit takes care of the charging through the USB port.

Component Name	Voltage range	Safety Level
Battery	3.7-3.9V	Vmin = 3.0V (for regulator to be effective)
Teensy Microcontroller	3.3V	V=3.3V+-0.1V
Wireless module	3.3-4V	V>3.3V

Table 1: Power Supply Specifications for Modules

Original Power Calculation:

We used the datasheet provided numbers [5] to calculate the estimated power consumption and total on-time of our device. This calculation is done using the MPR121 infrared ambient temperature sensor that we are considering in our IR design.

Teensy Maximum Power:

$$P = V * I = 3.3\ V * 185mA = 0.611\ W \qquad (EQ.\ 1)$$

Wireless Module Power (GM-205810-000):

$$P = V * I = 3.3\ V * 85mA = 0.281\ W \qquad (EQ.\ 2)$$

Total power efficiency can be estimated to be 80 % (based on the statistical efficiency curve [10]), the estimated maximum total power is:

$$Ptotal = (0.611+0.281) / 80\% = 1.115\ W \qquad (EQ.\ 3)$$

This indicates that a common 1500 mAh DC battery will support the whole circuit (in full-load operation) by a period of:

$$1.5\ Ah * 3.6V / 1.115\ W = 4.84\ hr \qquad (EQ.\ 4)$$

However, one should expect the cycle to be longer since this is the worst case scenario of having the whole circuit on full load.

III. Control

The control unit uses Teensy 3.2 microcontroller as the processor. The microcontroller and its peripheral module will provide an interface between the capacitive sensors array and computer, by taking in input data via analog input port. The high level algorithm created to map the input information to two dimensional direction outputs and other useful information. Eventually the output are sent via serial bus to communication module, connecting to any computers.

The high level algorithm is shown as follows:

```
Initialize input handling queue
Start calibration
For each data packet polling iteration:
      If not Gesture mode:
            Process first 8 inputs on the queue
            direction = calculate the angle()
            Compare direction info with last iteration
            Record direction change
      Else:
            // multi directional gesture click button
            If Gesture scrolling:
                  // scrolling feature
                  analyze left 3 and right 3 sensor patterns
                  Record as scroll up or down
            If gesture move selection:
                  Process next 9 elements on the queue
            If Gesture left click:
                  Record as left click
            If gesture right click:
                  Record as right click
```

Algorithm 1: Omni mouse software algorithm

In addition, we make few changes to our algorithm to eliminate the sensor noise.

The algorithm will handle the user input in three cases: Single input, two inputs and multiple inputs. The angle will be determined by either the position of single input source or the position difference between two concurrent inputs. If there are multiple inputs involved, the algorithm will run the two inputs algorithm on every pair of active sensors.

IV. Communication

The communication block consists of two connecting modules: Serial communication and Wireless Bluetooth communication. Both modules are considered peripheral modules to the control block. The USB module will also be part of rechargeable circuit, as previously mentioned. In our final design, we implement a python script to fetch packets from microcontroller and use pyautogui library to control the mouse.

Results

The requirements and verifications table can be found in Appendix A.

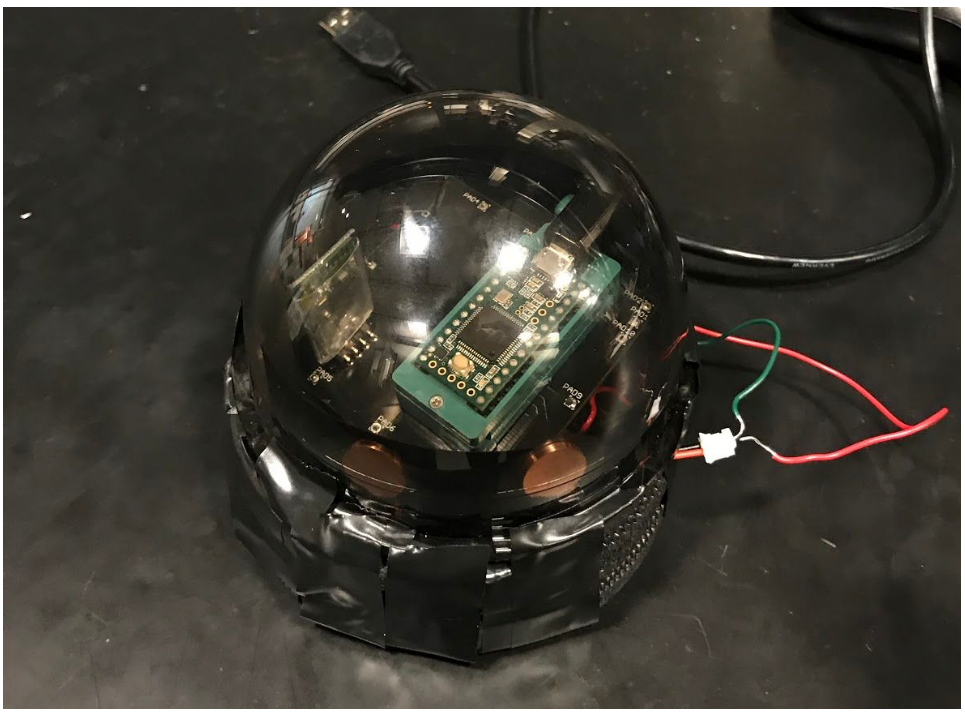


Figure 6: The OmniMouse

Challenges and Results

We ended up verifying the majority of our requirements, with the unachieved points being due to our python Bluetooth driver not being able to process the data fast enough. The Bluetooth driver was only capable of ~20Hz, while the wired mouse driver was capable of a satisfactory 62.5Hz. We speculate that the issue is due to the speed (or lack of) at which python scripts run. Originally we experienced major latency with the bluetooth driver, having both update speeds of under 5Hz, as well as an increasingly large delay with the movements as well; inputs wouldn't be registered until much later! To solve this, we investigated and tested several methods. We increased the Baud Rate of the Bluetooth Module (via hardware commands) to the maximum allowed 115200 from the initial 9600 Baud. We minimized our packet sizes (at one point reaching just 5 bytes per packet, which should have worked for even 9600 Baud). We limited the amount of packets transferred to the output buffer, as well as clearing the input buffer on the receiving end but still had noticeably slow operation of the mouse. Even though we optimized our code to the best of our ability, we were not able to break 20Hz, leading us to believe that the issue lies with the speed at which Python is able to process our data and run the corresponding mouse movements. This is an unfortunate result that we discovered late into the project, which led to us leaving it as it is to focus on other aspects of the project.

For the physical design of the mouse, we decided to go with an acrylic dome and a layered foam base. The foam was cut to fit the PCB and modules inside and was layered to provide the maximum insulation for the electrical components inside. The sensors themselves were copper discs made for us by the ECE Machine Shop. They are a ½ inch in diameter and ⅛ inch thick. They were mounted on the inner side of the acrylic dome using electrical tape. Wires were then connected to them from the PCB. Upon testing our physical design, we encountered a significant amount of interference that we found was coming from the circuit as a whole. Since the capacitance readings are quite sensitive, any interference from the Teensy, USB connection, power, or Bluetooth caused errors in our readings. To solve this, we insulated all possible points and reinforced the cables that were connected to the sensors.

For total power, It turns out that the final design draws an average current of 64mA from the DC supply that simulates the li-on battery. Thus, the charging period can be estimated as:

$$1200\text{mAh} / 64\text{mA} = 18.75\text{ hr} \qquad \text{(EQ.5)}$$

This is almost 5 times larger than our estimated period, which proves the power efficiency of our design.

In addition, the average power can be calculated as:

$$3.9\text{V} * 0.064\text{ A} = 0.25\text{ W} \qquad \text{(EQ.6)}$$

According to test experiment conducted by Microsoft, the wireless mouse on market active power draw varies from 11.7mA to 23mA [11]. It can be seen that the 64mA that our design draws is a little above the average products from the market. But we believe that it can be improved by using a more power-effective microcontroller and bluetooth module.

For power protection circuit, the main circuitry part works fine under test (Figure 7). The test results are shown on Figure 8 & 9. During this test, the cutoff voltage is set to be 2.0 V.

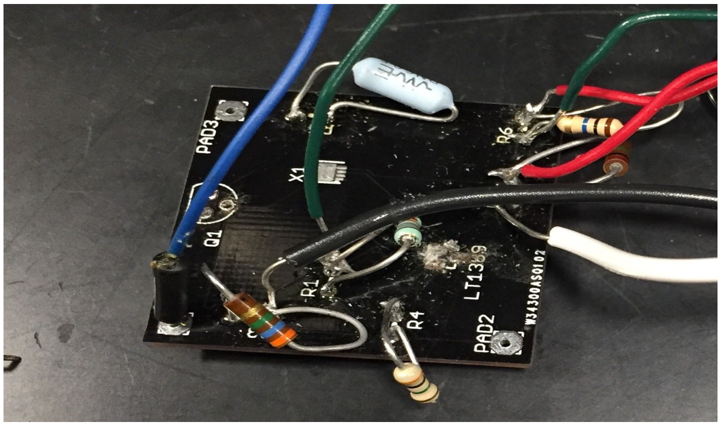


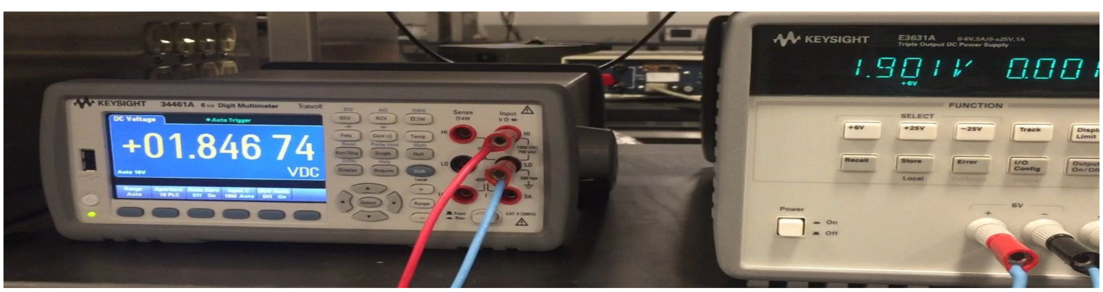
Figure 7: Soldered Power Protection Circuit

When the input battery voltage is 1.9V or below (shown on the right), the output of the op-amp is 1.85V (shown on the left), which is designed to be fed into the Pmos gate and turn it off.

When the input battery voltage is 2.0V or above, the output of the op-amp is 0.34V (shown on the left), which is designed to be fed into the Pmos gate and keep it on.

In this way, the protection circuit could instantaneously cut off the load from the battery when the battery voltage falls below the threshold number set by the bias resistors.

Unfortunately, all the Pmos we found could not fit with the main circuitry. This raises the practical concerns for electrical engineers in the real design process. It indicates that more practice and experience is needed to work in the professional industry.



it,

Figure 8: Power Protection Circuit Cut off the Load from the Battery

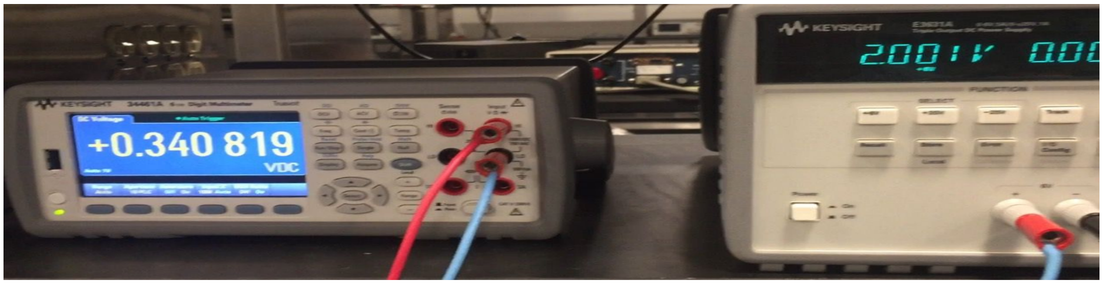


Figure 9 : Power Protection Circuit connects the Load with the Battery

Project Costs

Estimating the hourly salary of each team member to be equal to the average graduating ECE student [6] and the average work hours in a year to be 2087 [7], we calculate that the hourly salary of the team would be $112.84. Estimating an average of 2 hours worked per day over the entire time period, we arrive at a total of $18054.40 in salary for our team of three. Multiplied by 2.5, we get $45136 in labor.

Part Name	Quantity	Cost (Total)
Teensy 3.2 Microcontroller	1	$30
1200mAh Li-Ion Battery	1	$8
LD1117 3.3v Regulator	1	$2
Mouse Chassis	1	$8
Foam Sheets	2	$4
Capacitive Sensors	12	$6
HC-06 Bluetooth Module	1	$9
Micro-USB-B to USB-A	1	$1
USB-B Female	1	$0.5
USB-B to USB-A	1	$3
Max 1555 USB Charging Chip	1	$1.20

Linear Technology 1495	1	$3
Various Resistors, Diodes, Wires, Solder, etc	~	$5
Omron B3F-4055	1	$0.5
Total Cost:		$81.20

Table 2: Parts Cost Estimation

Our total cost for parts and labor are: $45136 + $81.20 = $45217.2.

Conclusion

Accomplishments

Ultimately, our mouse was able to accomplish the majority of our stated goals. The mouse works both wired and wirelessly on battery power. The mouse itself is quite responsive to user input, and is rather accurate. The mouse operates at a range of over 5 meters, and lasts an impressive 18.75 hours on the 1200mAh battery. On the other hand, the gesture input is difficult to activate, and the bluetooth mode did not meet our minimum requirements of 60hz. However, as an input device we have exceeded our own expectations. The actual user input and mouse movement feels enjoyable to most of the tested users, and has generated some amount of interest in the mouse. Though the device is still in its infancy, we feel there is a lot of promise behind the idea.

For the current imperfections of the mouse, we believe all the issues could be fixed if we had a bit more time. Many of our concessions in design were due to a lack of time before we had to have a working device. Printing new PCBs and ordering new parts take a certain amount of time that heavily eats into the development cycle.

Beyond the technical achievements of the mouse itself, we also achieved self growth as students, engineers, and innovators. We learned to manage our time and how to cooperate with teammates on the project. The whole experience has been invaluable, teaching us much more than we had originally expected.

Uncertainty

The main uncertainty behind our project is the bluetooth latency and the overall usability of the mouse. The current version is not robust enough to directly replace a standard user's mouse without some serious sacrifices. As such, we believe that it is necessary to address these concerns to meet the original goal of an ergonomic mouse. This uncertainty stems a lot from the gesture button input, which was not totally effective, as well as the overall durability of the mouse. Since the mouse is a prototype, these things are to be expected, but if the project is to be taken further, all of these concerns must be addressed.

Future work

To take the Omnimouse further, we expect to improve on four main parts:

1. Rewrite the bluetooth driver in C/C++ or change the bluetooth module. This will increase the rate at which the connected computer will update the mouse cursor, leading to less lag.

2. Design a sturdier base. The layered foam used was not sturdy enough for a final product. Ideally a 3D-printed or machined base would be ideal to give the mouse rigidity and durability.

3. Improve on the learning curve of the device. This can be addressed with a tutorial program that the user can either download, or is provided with the mouse itself. This way the user will have all the operational modes of the mouse explained.

4. A new PCB design. Since we changed some aspects of our project after the arrival of our PCB, we had to modify the boards and their connections. With a version 2 of the PCB we can get rid of those by implementing them directly into the PCB, as well as shrinking their size to fit better with the new base.

Ethics and Safety

Following the IEEE Code of Ethics [8], there are a few safety and ethical issues that we must address in our project. The first and foremost being the responsibilities as engineers to make decisions based on the safety and welfare of the public in mind. Thorough research on ergonomics of the human hand and the ideal positioning and angles of movement will be done for the design of the shell of the mouse. We used a dome shaped piece of acrylic for the mouse, but still need to do more long term research on the effects of the shape on the user. Furthermore, the movement as well as gestures must be non-fatiguing, or at least minimally fatiguing for the user to fit in with our objective. This will also need to be further tested to assure our design fits within our objective. These steps must be taken to ensure that the user is aware of any possible dangers of using the device. Since it is aimed to be an ergonomic solution, the device hopefully will be an improvement over existing products in this space. Since this is still a device that must be hand operated, the user should also be warned of the unavoidable possibilities of a device. Even though we aim to minimize harmful user interaction, the device will be connected to a larger electronic ecosystem that may be out of our control. One step we as the developers of this device can do, is to include warnings for using the computer for too long of a time, or in an extreme case, even offer a software lock that prevents further input until the user takes a break.

Since we utilize a li-ion battery in our mouse to allow for wireless connectivity through bluetooth, we must take battery safety into consideration. As outlined from the Battery Safety document located on the course website [9], Li-ion batteries are well known to be explosive if exposed to the wrong conditions. To combat this, we follow the provided guidelines, making sure to keep the battery temperature and voltage within safe ranges (3-4.2v). Additional measures, such as shielding and isolating the power circuitry will eliminate the risk of electric shock. These precautions should allow for the safe operation of our li-ion battery by itself, and in conjunction with our charging circuit.

Citations

[1] The World Bank. (2015) *Internet Users per 100 People* [Online]. Available, http://data.worldbank.org/indicator/IT.NET.USER.P2?page=6&cid=GPD_44

[2] Clay Scott. (2014) *Repetitive Strain Injury* [Online], Available, University of Michigan. https://web.eecs.umich.edu/~cscott/rsi.html

[3] Alan Hedge. (2011) *Ten Tips for Using a Computer Mouse* [Online], Available, Cornell University. http://ergo.human.cornell.edu/cumousetips.html

[4] Elegant Circuits. (2015) *USB Li-Ion Battery Charger* [Online], Available, http://elegantcircuits.com/2015/03/15/usb-li-ion-battery-charger/

[5] PJRC. (2012) *Teensy 3.2 Circuit Schematic* [Online], Available, https://www.pjrc.com/teensy/schematic.html

[6] United States Office of Personnel Management. (2016) *Computing the Hourly Rates of Pay using the 2087 Hour Divisor* [Online], Available, https://www.opm.gov/policy-data-oversight/pay-leave/pay-administration/fact-sheets/computing-hourly-rates-of-pay-using-the-2087-hour-divisor/

[7] University of Illinois Department of Electrical and Computer Engineering. (2016) *Salary Averages* [Online], University of Illinois, Available, https://www.ece.illinois.edu/admissions/why-ece/salary-averages.asp

[8] IEEE. (2017) *IEEE Policies 2017*, IEEE Inc. Available, https://www.ieee.org/documents/ieee_policies.pdf

[9] Jackson Lenz. (2016) *Safe Practices for Lead Acid and Lithium Batteries* [Online], University of Illinois, Available, https://courses.engr.illinois.edu/ece445/documents/GeneralBatterySafety.pdf

[10] Jukka Eskelinen and Kim Meyer. (2010) *Optimizing microcontroller power efficiency for ultra-low-power designs* [Online], Available, http://embedded-computing.com/articles/optimizing-power-efficiency-ultra-low-power-designs/#

[11] How much power does a wireless mouse use? [Online], Available, https://electronics.stackexchange.com/questions/131846/how-much-power-does-a-wireless-mouse-use

Appendix A

Input (20 pts)

Requirements and Points	Verification	Achieved
Microcontroller should process sensor data at at least 60Hz. (5 pts)	Set up a serial monitor on the receiving device, and read raw transmitted data for a period of time, checking that we get an average of at least 60 packets per second.	Yes
Gesture Button should send button input data. (5 pts)	Read transmitted data for button actuations. See if they perform the expected tasks on the test machine.	Yes
Sensors should provide accurate capacitive readings for user input. (5 pts)	Use a serial monitor to check the sensor readings from the device. Readings should be both consistent and varied depending on input (eg. higher value when finger placed directly on, lower value when partially on or near)	Yes
Sensors should record direction of user's hand input accurately. (5 pts)	Place hand along each of the sensor points as well as on two sensors at a time, and check the mouse movement relative to the input. Movement should give a similar angle as the input.	Yes

Table 1: Input Requirements and Verifications

Power (8pts)

Requirements and Points	Verification	Achieved
Charging Control PCB begins to charge the battery through USB when the battery's voltage drops below 3.0 V and stops charging when it is fully charged to 4.0V. (3 pts)	Feed the USB power lines from a computer to the charging PCB board. Verify the output voltage, which is to be fed into the Li-on battery, is 4.2 V (the desired charging voltage)	Partially

Battery lasts at least 4 hours before charging is needed (5pts)	Measure the power draw of the mouse using the DC power supply over a period of a minute, and then use the average voltage/current to calculate the power use and use that data to calculate total device-on time.	Yes

Table 2: Power Requirements and Verifications

Control (12 pts)

Requirements and Points	Verification	Achieved
Drivers should receive and process the mouse data packets at a speed of 60hz at least (6 pts total, 4 pts for USB, 2 pts for Bluetooth)	Small script to calculate the number of packets received for the wireless and the wired modes. For each, verify that the average packets processed per second is 60.	USB - Yes BT - No
The driver should eliminate sensor noise (6 pt)	Intentionally create at least two input noises (our hands, other materials) on sensor array for directional information. Intentionally create at least one input noise on left side and right side for scrolling. Test to see the resulting cursor movement. (Note: Design Choices)	Yes

Table 3: Requirements and Verifications

Communication (10 pts)

Requirements and Points	Verification	Achieved
The wireless communication should work with in the range of 1 meter (5 pts)	We will test the device at 1 meter, in a normal office and desk environment and home environment, the two main use cases for our device.	Yes
The USB mouse driver	Test mouse using a wired connection, then	Yes

should work in both wired and wireless modes. (5 pts)	unplug the cable to see if it still works in wireless mode.	

Table 4: Block Level Requirements and Verifications